1. PH.D. RESEARCH

My main research interests are adaptable distributed systems and applied security. I believe that the key problem in contemporary distributed systems is writing and automatically deploying distributed applications in dynamically changing heterogeneous networks, while ensuring that the user's QoS requirements are satisfied. Heterogeneous environments such as the Internet are formed by various devices (super-computers, PCs, hand-held devices) running diverse software (operating software, middleware) and being connected by links with different properties (available bandwidth, latency, security). This heterogeneity represents a problem that is either ignored or solved by increasing the complexity of distributed systems to levels that render manual management almost impossible.

One attractive solution would be to write simple distributed applications that express at a high level their efficiency, security requirements, and their malleability to environment changes and continuous user input, and create adaptable infrastructures that automatically and transparently deploy the applications with minimal user input. Unfortunately, no current system achieves these goals.

Some of the challenges are: (1) modeling the applications and the environment to capture complex application and environment properties and the relationships between them, (2) finding in a timely manner the best application deployment that satisfies both application and network constraints, and the user's QoS requirements (planning), and (3) securely and efficiently deploying the applications. These challenges become even harder when the heterogeneous environments are divided into multiple administrative domains, where each domain is allowed to define private credential and property namespaces. In this case, applications can be correctly and securely deployed only if there exists an appropriate translation of properties and cross-domain authorization of entities.

My dissertation work explores the thesis that by exposing qualitative and quantitative properties and relationships between component-based applications and heterogeneous environments, automatic deployment in resource constrained and dynamically changing environments becomes feasible. Feasibility implies that a valid application deployment is found if it exists, the automatic deployment process does not incur significant overhead compared to manually deployed solutions, and the QoS requirements specified by users are satisfied. In order to validate the idea of my thesis, I have developed a component-based framework (Partitionable Services Framework) which enables dynamic configuration and transparent adaptation of general component-based applications.

The Partitionable Services Framework (PSF) views applications as being dynamically built out of independent components that can be flexibly assembled to suit the properties of their environment, and facilitates on-demand transparent migration and replication of these components at locations closer to clients while still retaining the illusion of a monolithic service. A run-time system (Smock) is responsible for registering applications with the framework and serving incoming client requests. Whenever a client wants to access a service, the run-time system performs the necessary security checks (authentication and authorization), decides which level of service the client has the right to access, and asks a planning module (Sekitei) to compute a valid component deployment.

The high-level contributions of my research are: (1) defining suitable component and network models, (2) building a scalable planner which exploits the expressivity of the component model to efficiently find a valid plan, and (3) building a secure application deployment process. In addition, I have developed several mechanisms (PSF views and flexible cache coherence protocol) and used
others to support the secure deployment of distributed applications in heterogeneous environments composed of multiple administrative domains.

The models developed as part of PSF describe the application as a set of components, and the network as a set of nodes connected by links. In PSF, a component is defined by sets of implemented and required interfaces, where each interface is associated with application-specific properties. These properties are used to express component deployment conditions and effects. Similarly, nodes and links are associated with network-specific properties that specify the effects of the environment on the deployed application and vice-versa. However, the PSF models are richer than other models (CORBA, Web Services, OGSA) because they allow the specification of general qualitative properties (e.g. privacy, trust level) in addition to standard quantitative properties (e.g. CPU, bandwidth, latency) [1].

Given such a model, it now becomes possible to automatically decide which components make up the application and where they are located. Our solution develops an efficient planning algorithm based on classical AI techniques that takes advantage of our application and network description models. What differentiates this planning algorithm from similar algorithms is its ability to scale with the size of the network and the complexity of the application [2]. Recognizing that the chances of finding a valid plan increase with the diversity of the component set, PSF enriches the set of application components by creating at run-time new custom components (views) that have different properties than the original components.

Even after finding a suitable plan, the process of securely deploying this plan into the network is difficult because all entities (i.e. users, applications, nodes, links) belong to different administrative domains. We assume that each domain defines its own space of properties and credentials, and no domain has complete knowledge of all spaces. Thus, entities need to be authorized across domains, and the network properties need to be translated into application properties and vice-versa. In order to address both problems, PSF uses dRBAC, a PKI-based trust management and role-based access control system originally developed for expressing and enforcing security policies in coalition environments spanning multiple administrative domains. With dRBAC, domains can define roles and properties as credentials belonging to private spaces, and associate those credentials with each entity. Both problems of entity authorization and property transformation can be reduced to the problem of mapping a role/property local to one domain to a role/property local to another domain. Thus, PSF authorizes entities or transforms properties by building dRBAC credentials trees that solve the mapping problem [3].

Two additional challenges that must be addressed during deployment are fine-grained access control and maintaining consistency among several instances of the same component. PSF's solution to both problems relies on the notion of views. Views define the appropriate granularity for both access control and consistency. To the best of our knowledge, PSF is the first framework that provides caching services to component-based applications, and guarantees that the data copies held by different views are consistent. The cache coherence protocol implemented by PSF satisfies the consistency requirements of any component-based application (application-neutral) deployed in any configuration (flexible), while using application-specific information. The performance of this general cache coherence protocol is improved by allowing the application to specify (1) data properties to characterize the shared data, (2) triggers to indicate when updates need to be pushed or pulled between views, and (3) merge/extract methods to merge/extract updates from/into views and original components [4].
In order to evaluate the benefits offered by PSF, I have implemented a Java-based prototype of PSF and developed two proof-of-concept component-based applications: a security-sensitive e-mail application and an airline reservation system. The first represents security-aware applications, while the latter applications with dynamic and complex consistency requirements. These applications were deployed in environments that simulate several fast and secure domains connected by slow and insecure links, permitting me to analyze both the programming and the deployment processes. The conclusion was that PSF manifested the desired behavior: (1) the code added and the annotations required by PSF were minimal and very simple, (2) PSF adapted the deployment plans based on the state of the environment and the user-specified QoS requirements, (3) the actual time needed by PSF to deploy the applications was not prohibitive, and (4) the efficiency of PSF-deployed applications matched the efficiency of hand-crafted solutions.

2. Future Plans

For the future, I would like to pursue two different research avenues. The first one is extending the Partitionable Services Framework to more completely solve the key problem defined earlier. The second avenue is to use ideas introduced by PSF to solve interesting problems in other related areas (e.g. web caching, sensor networks).

**Extending PSF - Autonomic computing.** As mentioned before, the key level problem in distributed systems is that the complexity of deploying distributed applications in such environments is increasing and one possible solution is to automate as many functions as possible.

PSF solves a subset of the autonomic computing problems: modeling of applications and environments, efficient planning in resource constrained environments, authorizing entities in a loosely coupled federation of domains, and ensuring continuous and flexible application-level consistency for component-based applications. However, there are many other challenges that remain. First, the PSF application model should be extended to automatically infer the properties of component compositions based on the properties of the individual component. Similarly, the PSF linkage model should be enriched with semantics information, in the same spirit as emerging web services specifications such as BPEL4WS and OWL-S. Second, while the current implementation of PSF assumes a traditional network monitoring system like NWS or Remos, several interesting research problems need to be addressed to build a monitoring system that is secure (i.e. do not reveal state information on one domain to other domains), non-intrusive (i.e. some domains might not accept intrusive monitoring systems) and able to extract both qualitative (e.g. trust, privacy) and quantitative information (e.g. CPU, bandwidth) on the state of the environment. Third, several modules of PSF which rely on a centralized data store should be decentralized. For example, a decentralized planning algorithm would eliminate the assumption of complete knowledge on both the application and the network, and would allow each domain to compute and deploy partial plans on its nodes.

In the long run, my research goal is to extend the notion of adaptability by integrating the ideas of calibration and cost models into distributed applications. For many years, engineers have tried to formalize the notion of calibration for mechanical devices and create cost models that can be applied to various scenarios. I think that the same ideas can be also applied into distributed systems. Most distributed systems allow applications to adapt when the QoS requirements specified by users or applications are no longer satisfied. The challenge is to decide what is the adaptation cost and when the adaptation is satisfactory (e.g. how to calibrate the application).
Using PSF ideas in other systems. I also believe that there are many systems (e.g. sensor networks, pervasive systems, web caches), that have significant challenges that could be successfully addressed by ideas introduced by PSF. For example, the environments where pervasive applications are typically deployed can be described as extreme heterogeneous environments, where the devices and the links connecting them have highly different properties. In such cases, the behavior of each component can be influenced by the node properties, and thus the overall distributed application can be very sensitive to the deployment plan. The challenge is no longer in finding the first plan that satisfies the user’s QoS requirements, but finding an optimum plan (e.g. minimum resource consumption, minimum response time). Similarly, sensor networks like IrisNet can be regarded as distributed systems where small services can be deployed on nodes and executed to gather and process sensor data. In this case, the challenge is deploying senselets on appropriate sensor nodes, if the deployment depends on properties such as the sensor type, the data production rate, the required available CPU and memory. Recognizing that this problem is similar to the PSF problem, one can extend the PSF models to capture properties and use the planner to find a valid plan. In addition, sensor networks can also use the notions of properties, triggers, and merge/extract methods introduced in the PSF consistency protocol to manage the consistency requirements specified by sensing applications.

REFERENCES